

1/PRTS

METHOD OF GENERATING ELECTRONIC KEYS FOR A
PUBLIC-KEY CRYPTOGRAPHY METHOD AND A SECURE PORTABLE
OBJECT USING SAID METHOD

5

The invention relates to a method of generating electronic keys for a public-key cryptography method. It also relates to a secure portable object using the method.

10

The invention relates more particularly to the generation of keys for an RSA-type cryptography system and to their storage on a secure object with a view to using them in an application requiring security.

15

The invention applies most particularly to secure objects which do not have a large memory resource, such as an electrically programmable memory, or powerful calculation resources, as is the case for chip cards.

20

One application of the invention is electronic commerce using a mobile telephone. In this context, the keys may be on the SIM card of the telephone.

5 It is provided that some application programs use such keys to carry out a transfer of confidential data, in an electronic commerce context for example. Hereinbelow, it will be considered that these applications are provided by a service provider.

10

Moreover, it is known that, in order to guarantee the integrity of the key, a certificate provided by a certification authority is usually associated therewith.

15

Among these public-key cryptography methods, the text below deals with the RSA (Rivest Shamir and Adleman) cryptography protocol. This protocol uses a step of generating large prime numbers which takes up a lot of calculation time and memory space.

20

It will be recalled that this RSA cryptography protocol allows information encryption and/or authentication between two entities and/or the electronic signing of messages.

25

The RSA cryptography protocol is used most frequently because it has properties which allow it to be used both for encryption and for signature generation.

30

To do this, the RSA cryptography system comprises a "public" algorithm which performs the encryption or signature verification function and a "private"

algorithm which performs the decryption or signature generation function.

5 The security thereof is based on the difficulty
in factoring a public integer N of large size which is
the product of the two secret prime numbers p and q of
large size, the pair (p, q) being used in the
calculation of the secret key d which is used by the
decryption function or by the signature calculation
10 function.

In order to better understand the problem which
will be discussed below, a summary is given of the
parameters used in an RSA cryptography scheme:

15 1) the public exponent e :

This is specific to an application and is
provided by this application. It is thus common to all
the users of this same application.

20 2) the parameters p and q :

These are generated from a calculation which
takes up a lot of time. They usually have the same
length (same size). This length is conventionally 512
bits. In order to increase security, this length may
extend from 512 bits to 2048 bits, 2048 bits being
25 envisaged for the future.

3) N is the public modulus and is calculated
from the following relation:

$$N = p \cdot q$$

30 The key of the algorithm is said to be of
length ℓ when the public modulus N is of length ℓ . This
length is set by the application (or service provider).

4) the parameters e and N form the public
key.

5) the private key d is calculated from the following relation:

$$d = 1/e[\text{mod}(p-1)(q-1)]; (1/e=e^{-1})$$

i.e. $ed = 1 \text{ [(mod ppcm}(p-1, q-1))]$; ppcm is
5 the smallest common multiple,

the secret parameters are formed by the triplet (d, p, q) .

6) the "normal" form of the private key is:
 (d, N) .

10 6) the CRT (Chinese Remainder Theorem) form of the private key is:

in this case the private key comprises 5 parameters:

15 p, q
 d_p with $d_p = d \text{ mod}(p-1)$
 d_q with $d_q = d \text{ mod}(q-1)$
 I_q with $I_q = q^{-1} \text{ mod} p$.

20 The principle of generating a key according to the RSA scheme therefore consists, as can be seen, in generating a private key d from a public exponent e (or public key) which is set by the application, the parameters p, q being generated such that $p \cdot q = N$, the length ℓ of N being fixed.

25 When a number of applications are provided, each service provider provides its public exponent e and the length of the public modulus N , so that the corresponding private key d can be generated.

30 Thus, carrying out an RSA key calculation requires knowledge of the public exponent e and of the length ℓ of the key of the algorithm, that is to say the length of the modulus N . With the input data e and ℓ ,

there remains to be generated the pair of prime numbers p and q such that the latter satisfy the following conditions:

- (i) $p-1$ and $q-1$ are prime numbers with e and
- (ii) $N = p*q$ is an integer of length ℓ .

These constraints take up a lot of calculation time.

In this respect, it will be recalled that the generation and storage of keys for portable objects such as chip cards are today carried out in two ways as follows:

According to a first way, the calculation of an RSA key is carried out on a server in order to benefit from considerable calculation power. For more security, a certificate is required which is downloaded with the key within the secure object during its personalization phase.

This solution has two drawbacks:

- on the one hand, despite the relatively secure context of the personalization, theft or duplication of the key may occur on account of its transfer from the server to the secure object, and
- on the other hand, each key is loaded into the object in an initial personalization phase, which requires that a maximum number of keys be provided in each object in order to be able to anticipate future requirements.

In practice, there are stored in the portable object sets of keys and certificates corresponding to

each application that is likely to be used, without knowing whether these keys will actually be used at a later date. A large amount of memory space is needlessly taken up. For example, 0.3 Kbytes are
5 required for an RSA key having a modulus of 1024 bits, whereas cards at present have at most 32 Kbytes of programmable memory. Moreover, a large number of certificates is purchased from the certification authority, and this is expensive.

10

Last but not least is the drawback that it is not possible to add new keys as and when new applications are envisaged.

15

According to a second solution, the calculation may be carried out within the secure object. This overcomes the first drawback of the above solution but creates a large amount of processing within the secure object, which has a small calculation capacity.

20

When the generation of an RSA key is carried out by a portable object such as a chip card, if the imposed length of the RSA key is 2048 bits, the calculation then takes 30 seconds with a powerful
25 algorithm.

25

Although this calculation time is acceptable for some applications since the RSA keys are generated just once for a given application, it is not satisfactory
30 for mobile telephony services (GSM for example) since this operation is renewed each time the SIM card is changed and a larger number of keys has to be provided in order to meet the requirements of different applications.

30

Due to a need for considerable calculation resources, the keys are always created during the personalization phase from the public exponents e provided by the various service providers. This calculation step cannot be carried out subsequently since it would paralyze operation of the object.

In practice, this calculation is not carried out by the card. This is because this calculation is lengthy and it could slow down the personalization phase, and also its duration varies and could prove to be incompatible with the personalization methods of the chip cards.

Moreover, this solution still has the second drawback of the preceding solution, namely the need for memory space.

The object of the present invention is to solve these problems.

More precisely, the object of the invention is to solve the problem of the calculation complexity associated with managing the generation of keys and also the problem of the lack of flexibility due to the initial and definitive storage of a large number of keys and certificates during the personalization phase.

To this end, one object of the present invention relates to a method of generating electronic keys d for a public-key cryptography method using an electronic device, mainly characterized in that it comprises two separate calculation steps:

Step A

- 5 1) calculating pairs of prime numbers (p,q) or values representative of pairs of prime numbers, this calculation being independent of knowledge of the pair (e,l) in which e is the public exponent and l is the length of the key of the cryptography method, l also being the length of the modulus N of said method,
- 10 2) storing the pairs or values thus obtained;

Step B

- calculating the key d from the results of step A and knowledge of the pair (e,l) .

15

 According to a first variant, step A-1) consists in calculating pairs of prime numbers (p,q) without knowledge of the public exponent e or of the length l of the key, using a parameter Π which is the product of small prime numbers. In this way, pair (p,q) obtained in step A has a maximum probability of being able to correspond to a future pair (e,l) and will make it possible to calculate a key d when step B is carried out.

25

 According to another variant which depends on the preceding variant, the calculation A-1) also takes account of the fact that e has a high probability of forming part of the set $\{3, 17, \dots, 2^{16+1}\}$, and for this use is made in the calculation of step A of a seed σ which makes it possible to calculate not pairs (p,q) but a representative value referred to as the image of the pairs (p,q) .

30

The storage A-2) then consists in storing this image. This makes it possible to gain memory space since an image is smaller than a prime number p or q , for example 32 bytes compared to 128 bytes.

5

According to a third variant, a calculation of pairs (p, q) is carried out for different probable pairs (e, l) . In practice, the parameter Π will contain the usual values of e , for example 3, 17.

10

According to a fourth variant, step A-1) comprises an operation of compressing the calculated pairs (p, q) and step A-2) then consists in storing the compressed values thus obtained.

15

Step B comprises the verification of the following conditions for a given pair (e, l) :

- (i) $p-1$ and $q-1$ are prime numbers with e and
- (ii) $N=p*q$ is an integer of length l .

20

According to one preferred embodiment, step A-1) comprises the generation of a prime number q , the selection of a lower limit B_0 for the length l_0 of this prime number that is to be generated, such that $l_0 \geq B_0$, for example $B_0 = 256$ bits, and it also comprises the following sub-steps:

25

1) calculating parameters v and w from the following relations and storing them:

$$v = \sqrt{2^{l_0-1}/\Pi}$$

30

$$w = 2^{l_0}/\Pi$$

in which Π is stored and corresponds to the product of the f smallest prime numbers, f being selected such that $\Pi \leq 2^{B_0}$,

2) selecting a number j within the range of integers $\{v, \dots, w-1\}$ and calculating $\ell = j \Pi$;

3) selecting and storing a prime number k of short length compared to the length of an RSA key within the range of integers $\{0, \dots, \Pi-1\}$, (k, Π) being co-prime;

4) calculating $q = k + \ell$,

5) verifying that q is a prime number, if q is not a prime number then:

a) taking a new value for k using the following relation:

$k = a \cdot k \pmod{\Pi}$; a belonging to the multiplicative group Z^*_Π of integers modulo Π ;

b) repeating the method from sub-step 4).

Advantageously, step B comprises, for a pair (p, q) obtained in step A and a given pair (e, l) :

- verifying the following conditions:

(i) $p-1$ and $q-1$ are prime numbers with e and

(ii) $N = p \cdot q$ is an integer of length ℓ ,

- if the pair (p, q) does not satisfy these conditions:

- selecting another pair and repeating the verification until a pair is suitable,

- calculating the key d from the pair (p, q) obtained at the end of this verification.

The invention also relates to a secure portable object able to generate electronic keys d of an RSA-type cryptography algorithm, characterized in that it comprises at least:

- communication means for receiving at least one pair (e, l) ,

- a memory for storing the results of a step A consisting in:

5 calculating pairs of prime numbers (p,q) or values representative of pairs of prime numbers, this calculation being independent of knowledge of the pair (e,l) in which e is the public exponent and l is the length of the key of the cryptography method, l also being the length of the modulus N of said method,

10 - a program for implementing a step B consisting in:

 calculating a key d from the results of step A and knowledge of a pair (e,l) .

15 The secure portable object also comprises a program for implementing step A, steps A and B being separate in terms of time.

20 The secure portable object may consist of a chip card.

25 Other features and advantages of the invention will emerge clearly from reading the description which is given below by way of non-limiting example and with reference to the single figure which shows a diagram of a system for carrying out the method.

30 The rest of the description is given within the context of application of the invention to a portable object of the chip card type, and for simplification this will be referred to as a chip card.

 According to the proposed method, the generation of keys is carried out in two separate steps.

The first step A comprises a calculation of pairs of prime numbers (p,q) or of values representative of pairs of prime numbers referred to as an image.

5

The pairs (p,q) obtained are stored.

This calculation is complex and it is even more complex if a conventional prime number generation algorithm is used.

10

It is proposed here that this calculation be carried out independently of knowledge of the pair (e,l) .

15

As will be detailed below, one preferred embodiment for carrying out this step makes it possible to simplify the calculations and to limit the memory space needed to store the pairs (p,q) obtained, by storing an image of these pairs.

20

The second step B comprises the calculation proper of the key d from the results of step A and knowledge of the pair (e,l) .

25

This calculation comprises, for a pair (p,q) obtained in step A and a given pair (e,l) :

- verification of the following conditions:

- (i) $p-1$ and $q-1$ are prime numbers with e and
- (ii) $N=p*q$, this number must be an integer of length l ,

30

- if a pair (p,q) does not satisfy these conditions, another pair is selected and the

verification is repeated until a pair is suitable among the pairs obtained in step A.

5 - it is then possible to proceed with the calculation of the key d from the pair (p,q) obtained at the end of this verification.

10 The first step, which corresponds to a relatively complex calculation compared to the second step, may be carried out by an element other than the chip card, for example by a server. In this case, the results of the calculation of this first step may be loaded onto a chip card during personalization.

15 The calculation of step A may also be carried out by the card itself at any given instant which does not disturb the user of this card. For example, this calculation may be carried out during personalization of the card or subsequently.

20 In practice, during use of the card, in order to obtain a service, if a private key is required then the public key is provided by the service provider (possibly remotely if it is not already stored in the card) in order to generate the private key. This
25 generation step (calculation step B) is carried out rapidly by the card.

30 It can be seen therefore that new applications which require the calculation of a private key d can be provided for a card.

 It can also be seen that there is no need to associate a certificate with the pairs (p,q) since they are not associated with a private key.

Thus, the generation of a private key can be carried out on board, that is to say by the card itself, with a 10-fold gain in execution time compared to the key generation methods known to date.

In the text which follows, a description will be given of one preferred embodiment for carrying out step A. This embodiment is particularly advantageous for use on board a chip card since it makes it possible to optimize both the memory space and the calculation time.

Firstly, in order to ensure that $N=p*q$ is an integer of ℓ bits, p is selected within the range:

$$\left[\sqrt{2^{2(\ell-10)-1}}, 2^{\ell-10} - 1 \right]$$

And q is selected within the range:

$$\left[\sqrt{2^{2\ell_0-1}}, 2^{\ell_0} - 1 \right]$$

For ℓ_0 between 1 and ℓ .

Thus, $\min(p)\min(q)$ is between $2^{\ell_0}-1$ and N , and $\max(p)\max(q)$ is between N and 2^ℓ as required.

In this way, condition ii) mentioned above is reduced to searching for prime numbers within the range:

$$\left[\sqrt{2^{2\ell_0-1}}, 2^{\ell_0} - 1 \right]$$

The proposed solution makes use of the parameter Π . This parameter Π is the product of small prime numbers among which there may be found in particular 3, 17, 2^{16+1} , which prime numbers are usually used as public exponents. Thus, the probability that a pair (p,q) will correspond to a given future pair (e,l) , which is already very high, rises even further when Π comprises such values.

The f smallest prime numbers are selected, f being selected such that $\Pi_i p_i \leq 2B_0$, B_0 is the lower limit selected for l_0 . For example, B_0 may be selected to be equal to 256 bits.

Π is equal to the product: $2.3....191$ and is less than 2^{256} .

This value Π may then be stored in the card for example as a constant in the program read-only memory.

The first phase of the method consists in generating and storing a prime number k of short length compared to the length of an RSA key within the range of integers $\{0, \dots, \Pi-1\}$, (k, Π) being co-prime, that is to say not having a common factor.

The second phase then consists in constructing, from this number k , the first candidate q which satisfies the condition of being co-prime with Π .

If this first candidate does not satisfy this condition, then it is updated, that is to say another candidate is selected, until a value of q which does satisfy the condition is found.

5 A description will now be given of the various steps of the algorithm for generating a prime number that is used in the calculation of an RSA key according to the invention.

10 The proposed algorithm works regardless of the length l_0 given for the prime number q that is to be generated.

15 The generation of the prime number p is identical; all that is required is to replace q with p in the steps which will be developed and to replace l_0 with $l-l_0$.

20 After having set the limit B_0 , the unique prime numbers v and w which satisfy the following conditions are calculated:

$$\begin{aligned} \sqrt{2^{2\ell_0-1}} \leq v\Pi \leq \sqrt{2^{2\ell_0-1}} + \Pi \\ 2^{\ell_0} - \Pi \leq w\Pi \leq 2^{\ell_0} \end{aligned}$$

25 This means calculating v and w by the following relations:

$$\begin{aligned} v &= \sqrt{2^{2\ell_0-1}} / \Pi \\ w &= 2^{\ell_0} / \Pi \end{aligned}$$

30 Then, having taken k belonging to the multiplicative group $Z^*\Pi$ of integers modulo Π , the first candidate q is constructed such that

$$q = k + j\Pi \text{ for any } j \text{ belonging to the range } [v, w-1].$$

Since k matches $Z^*\Pi$, the probability of having a prime first candidate q is high. If this is not the case, k is updated by taking k equal to $ak \pmod{\Pi}$, a belonging to the group $Z^*\Pi$, and the method is repeated until a value of q which corresponds to a prime number is found.

One way of testing the primality of a number is for example to use the Rabin-Miller test.

10

The various steps of the proposed algorithm are specifically as follows:

1) calculating parameters v and w from the following relations and storing them:

$$v = \sqrt{2^{2^{l_0}-1}}/\Pi$$

$$w = 2^{l_0}/\Pi$$

in which Π is stored and corresponds to the product of the f smallest prime numbers, f being selected such that $\Pi \leq 2^{B_0}$,

2) selecting a number j within the range of integers $\{v, \dots, w-1\}$ and calculating $\ell = j \Pi$;

3) selecting and storing a prime number k of short length compared to the length of an RSA key within the range of integers $\{0, \dots, \Pi-1\}$, (k, Π) being co-prime;

4) calculating $q = k + \ell$,

5) verifying that q is a prime number, if q is not a prime number then:

a) taking a new value for k using the following relation:

$k = a \cdot k \pmod{\Pi}$; a belonging to the multiplicative group $Z^*\Pi$ of integers modulo Π ;

b) repeating the method from step 4);

6) recording a , k' j in order to use them to find q and then making use of q during a subsequent calculation to generate an RSA key.

5 Instead of storing the value of q , the method will advantageous proceed as described below.

10 A simple manner for carrying out this algorithm may consist, for each length of RSA key envisaged, in storing the values of k and j so as to reconstruct q .

15 Rather than selecting a random number j as indicated in step 2), another embodiment may consist in constructing j from a short random number.

20 For example, a number having a length of 64 bits is taken, which is referred to as the seed and is denoted σ . This seed is then taken as the input value of a pseudo-random number generator PRNG, which will make it possible to generate j .

j is then defined as $\text{PRNG}_1(\sigma) \pmod{(w-v)+v}$.

25 This embodiment makes it possible to considerably reduce the requirements in terms of memory space since only the values of σ and k have to be stored in the EEPROM memory. The value of Π is in the read-only memory (in the calculation program).

30 It is possible to further reduce the requirements in terms of memory space by acknowledging that: if $k_{(0)}$ is the first value of k belonging to the group $\mathbb{Z}^* \Pi$, then the prime numbers generated have the form:

$$q = a^{f-1} k_{(0)} \pmod{\Pi} + j \Pi$$

f being the number of failures of the test in step 4).

5 This value $k_{(o)}$ which belongs to the group $Z^*\Pi$ may be easily calculated from a short random seed, such as σ for example, and using the Carmichael function of Π^2 denoted $\lambda(\Pi)$.

10 Using this function, it is possible to express $k_{(o)}$ by the following relation:

$$k_{(o)} = [\text{PRNG}_2(\sigma) + b^{\text{PRNG}_3(\sigma)} (\text{PRNG}_2(\sigma)^{\lambda(\Pi)} - 1)] \pmod{\Pi}$$

b being an element of order $\lambda(\Pi)$ belonging to $Z^*\Pi$.

15 These two embodiments make it possible to reduce the requirements in terms of memory space since in this case all that will have to be stored is the value of the seed σ and various values of f for the desired key lengths.

20 For RSA keys having a modulus of greater than 2048 bits, the numerical experiments that have been carried out by the inventors show that f is equal to 2^8 . This means that f may be encoded on 1 byte, i.e. 8
25 octets.

30 By way of example, for generating RSA keys having a length ranging from 512 to 2048 bits with a granularity of 32 bits, there are 49 possible key lengths. It is thus necessary to store on the card one byte, i.e. 8 octets, corresponding to the value of σ . It is also necessary to store the values of f for the prime numbers p and q, i.e. $2 \times 49 = 98$ octets. This makes a

total of 106 bytes, i.e. 848 bits, in the EEPROM memory.

5 A final embodiment that makes it possible to reduce the memory space consists in storing in the calculation program, that is to say in the program memory, a number of values of Π and the corresponding values of $\lambda(\Pi)$ for various envisaged key lengths. It may be noted that a large value of Π leads to the
10 smallest values for f .

As has been seen above, the prime number q generated according to step 4) by the algorithm which has just been described satisfies the condition:

$$15 \quad q = a^{f-1} k_{(o)} \bmod \Pi + j \cdot \Pi$$

If e divides Π , q can be expressed by the following relation:

$$20 \quad q = a^{f-1} k_{(o)} \bmod(e)$$

In order that condition (i) mentioned at the start of the description is satisfied, a must be selected such that $a \equiv 1 \pmod{e}$ and k must be forced to be different from $1 \pmod{e}$.

25 The prime number q obtained thus satisfies the relation $q = k_{(o)}$ different from $1 \pmod{e}$.

The generation of the prime number p is identical
30 except that q is replaced with p in the steps which have been developed and l_0 is replaced with $l-l_0$.

As has been mentioned, the program that implements the method of the card does not a priori

need to know the public exponent e . This exponent can therefore be provided at any moment by an application loaded into the card.

5 However, it is known that, for most applications (more than 95%), the values of e that are used are the values $\{3, 17, 2^{16}+1\}$.

10 In order to cover the greatest number of applications, a will preferably be selected such that $a \equiv 1 \pmod{\{3, 17, 2^{16}+1\}}$ and $k_{(0)}$ must be different from this value: $1 \pmod{\{3, 17, 2^{16}+1\}}$.

15 For example, the prime number $R = 2^{64}-2^{32}+1$ is selected as a possible candidate for a , with the proviso that the greatest common divisor of Π and R is equal to 1.

20 The required condition for $k_{(0)}$ may be obtained by the Chinese Remainder Theorem.

25 As already mentioned, another alternative may consist, in respect of step A-1), in calculating pairs of prime numbers (p,q) for various probable pairs (e,l) .

30 In conclusion, the invention proposes a method consisting of two separate steps, in which the second step, which is very quick compared to the known solutions, can be carried out in real time. This method also takes up a relatively small amount of memory space.

Moreover, there is no limit in terms of new applications not provided at the time of personalization of the card.